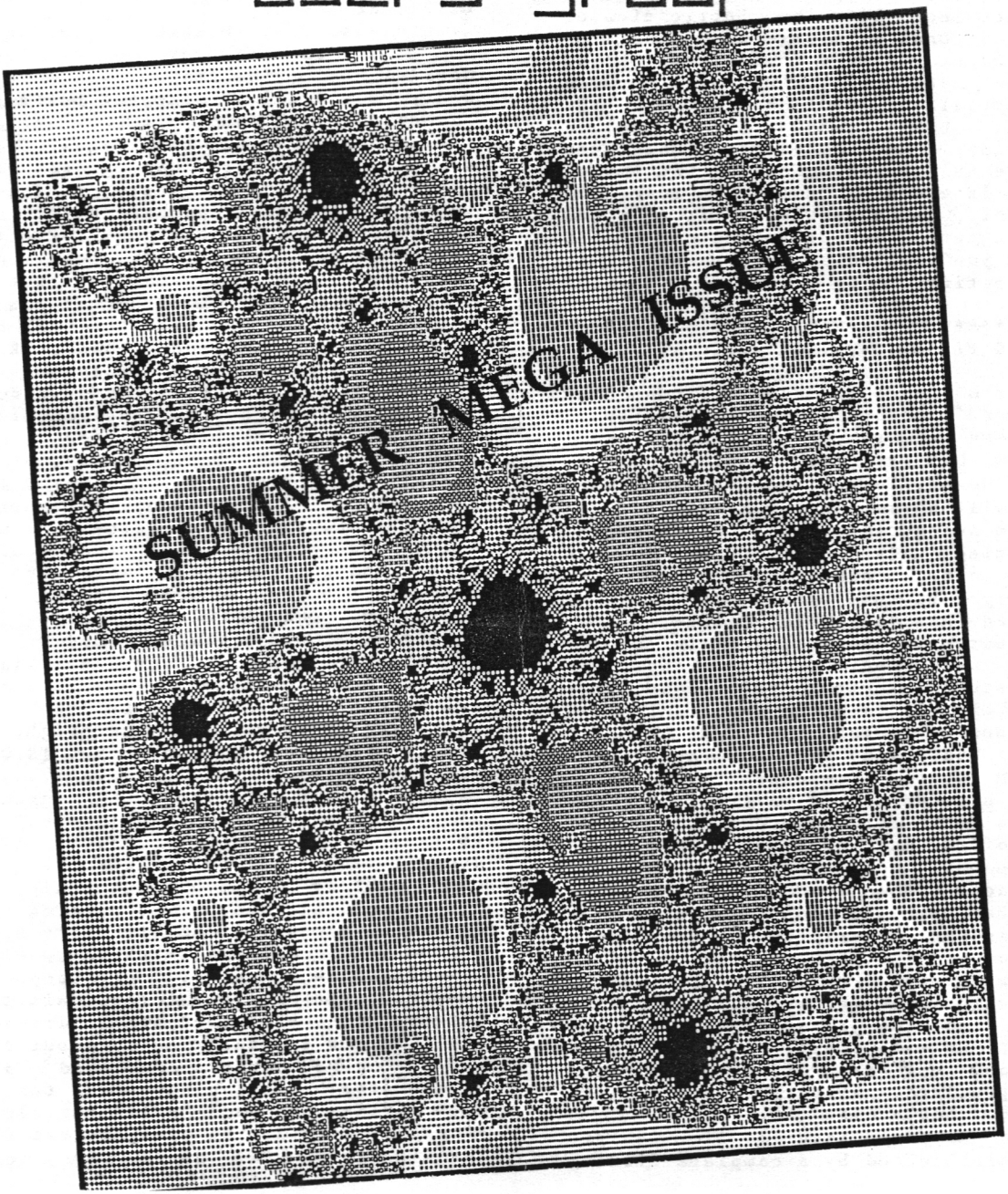
Summer/88

ZX-Appeal

vancouver sinclair users group



THIS ISSUE.....

The dog days (why not the CAT days?) of summer. If you're looking for a break from the sweltering heat, open a cool one and go down into the coolness of your basement and relax with that special 'fun' program you enjoy.

The 'SUMMER MEGA ISSUE' has something, I hope, for everyone. Harvey gives us another look into the intricacies of the QL; Gerd is along with some tips for using Wilf's 32k NVM 'Delta Device' more fully; Vince pops up with a tutorial on Boolean Logic - the basics behind how our machines function; and Fred left us some gems before he ventured East. The summer issue is where I get the space to reprint those articles from other newsletters and other sources that I think you'll find especialy helpful or interesting. So without further ado...

****

BITS & PIECES.....

...a few weeks ago Fred N. popped up at Gerd's house to say "see ya later" to some members of the group. Fred was on his way from Nelson to Ottawa where he has located honest employment - he will be setting up a computerized office accounting package for a friend utilizing, wait for it, TS machines!!! Fred loaned his entire software library to the group as well as donated his sizeable collection of newsletters, magazines, articles and other neat stuff to the group. All this stuff is being cataloged and will be available to group members through the various librarians. Best of luck, Fred. Don't stay away too long.

...CTM quits. Computer Trader Magazine started out as a small regional Buy & Sell for computers and became one of the mainstays for Packet Radio and hobby computers - mainly TS machines. When TS Horizons packed it in, Chet decided to take over the outstanding subscription obligations - at no pay! The reality of the economics of trying to publish a specialty magazine while keeping the price as low as possible finally caught up with Chet. A big "thanks and good luck" from all of us at VSUG. We appreciated everything you did to help our hobby along.

...the May/June issue of Time Designs was highlighted by a complete listing of all the active TS User Groups and all the dealers and suppliers still supporting our machines. Would you believe 55 active groups and 94 dealers and suppliers? Not bad for an 'orphan' long forgotten. If you want to obtain this list, Tim advises he had a larger than usual print run so just write for a copy - I'm not sure of individual copy price but why not just take out a subscription starting with that issue!

...the same Time Designs issue also had a ranking of User Group Newsletters. The ranking was made by a 3 person panel of computer buffs not connected with TS machines. Congrats to the N/L of the Las Vegas Users Group. The 'Hacker' was judged top o'the heap. Editor Steve graciously replied that the newsletters of all the groups were winners just for still being here. Where did ZXAppeal rank? Tied for 2nd with the Capital Area TS groups N/L! This is a great honour for ZXApeaal as I think the CATS N/L is really super. Since this ranking appeared we picked up 8 new members.

...were you at the 3rd Annual International/Great NW Timex/Sinclair Mini-Fair? If not you missed a fun time. Jay, Eric, Neil, Harvey, Wilf, Kenton, Jim, Tim, Rod, Ramine, & Ken made it. Wilf and Harvey each gave a seminar. The number of attendees was not as was expected but the dealer tables and the seminars were great. Some super bargains were to be had near the end of the Fair. The declining user base and the non-central location were certainly factors in the attending numbers but the enthusiasm displayed by the 150 who were there was just as bright as ever. VSUG held a draw for a TS1000 and rampak - and the winner was..... Michael Carver of the CCATS group. Mike is known for his articles in Time Designs on things QLish. VSUG also contributed 5 doorprizes of a complimentary 6-month membership in the group. All those who entered the draw will receive a complimentary copy of the SUMMER MEGA ISSUE of ZXAppeal.

...George Whitham of A+ Computer Response, the QL distributor, said the reponse to the recent A+ advert for $75 QLs was so great he sold out the 400 he had left and had 300 back-orders. George reported that a West German company is in the final stages of negotiation with Amstrad for the rights to the QL so they can

manufacture a QL for the W/German market. This model will include a 3.5"DD in place of the M/Drives. Apparently the demand for QLs in W/Germany is incredible and 10,000 machines could be sold tomorrow - if they were available.
...Nigel Searle was at the ´Fair and gave a seminar on the Z88.(He couldn´t have done it without crucial assisstance from our own ex-prez Ken.) Nigel has been with Clive Sinclair since 1972 and was able to entertain all with stories of the bad old early years. Nigel is now involved with the entry of the Z88 into the U.S. and Canada. Yes, Canada. I´m assured we should be able to see it ´in the stores soon´.
****

RENEWING MEMBERS:

Hilda McKinnon, Kevin Kearney
Brad Thomas, Tim Stoddard
Jon Kaczor, Kenton Garrett
Dave Noordhoff

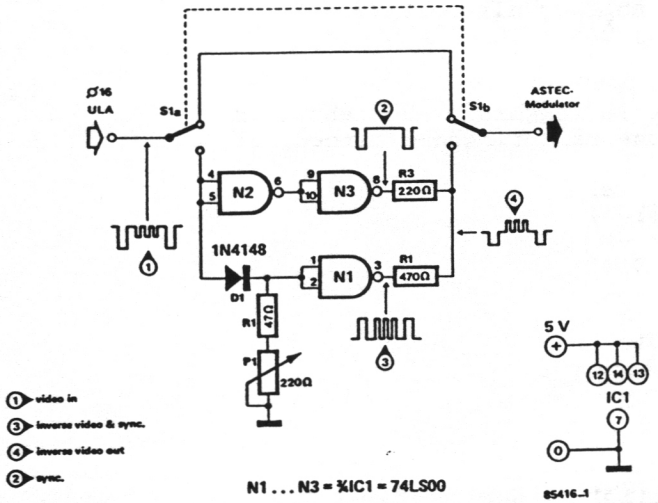
NEW MEMBERS:

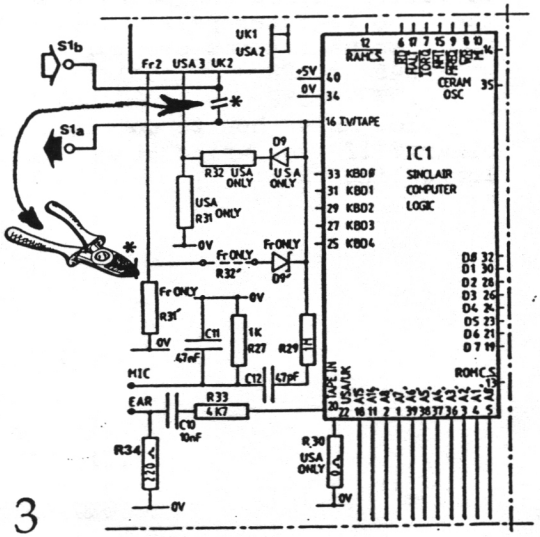
Eric Tsoi, Vancouver, BC
Charles Byler, Ft. Riley, Kansas
Van Vangor, Island Falls, Maine
Don Myers, Chula Vista, Calif
Bill May, Jr., Yeadon, PA
Jim Kopisch, Oklahoma City, OK
Don Lambert, Cedar Rapids, Iowa
Seward Warner, Liverpool, NY
Chris Crawford, Gilmer, TX

BE SURE YOU RENEW NOW IF YOUR LABEL SO INDICATES
****

simple video inverter for ZX81

The inverter must be connected before the TV modulator in the ZX81. Switch S_1 enables bypassing of the inverter when inversion of the picture is not required. The composite video signal is inverted by gate N_1. Gates N_2 and N_3 separate the sync signal from the input: the sync signal is then available at the output of N_3 at a level of 5 V_{pp}. The inverted video signal and amplified sync signal are then added again, resulting in an inverted video signal with the sync signal in the correct position and at the right level. Preset P_1 serves to adjust the contrast.
The circuit can be constructed on a piece of veroboard so small that it can easily be added in the ZX81 case. The power supply can be taken from IC_1 in the ZX81: +5 V at pin 40 and earth (0 V) at pin 34.





PLAYING WITH ELECTRICITY

- July 12/88
- Harvey Taylor

Back in the good old bad days when I was programming on an 8 bit CPU, I used to hear talk of multiplication routines and was rather mystified just how this wonder was worked in binary. When I got into 68xxx programming, there was, lo & behold, a multiply instruction which removed any need to bother to figure out how binary multiplication might work ... until I ran into the need to multiply numbers greater than 16 bits. I started playing with multiplication lately to enhance my Mandelbrot generating program and thought you might find the topic of interest.

One of the first methods of multiplying a machine language programmer runs into is shifting. Let us say you wanted to calculate 2 times some small number. The quickest and easiest way to do this is just to add it it to itself. eg. ADD A,A in Z8Ø or ADD DØ,DØ will leave double the initial value in the register. So will shifting the number one position to the left. eg. SLA A or LSL #1,DØ These two methods have the same effect. The addition method is faster if you are counting cycles.

It is relatively straightforward to multiply by a small integer number. Let us say you want to calculate 9 times some other small number. You could do it something like this.

```
LD  B,A      ; SAVE A COPY
SLA A        ; X2
SLA A        ; X4
SLA A        ; X8
ADD A,B      ; X9
```

Similarly some odd numbers can be constructed of sums of shift products. eg. 13n = 8n + 4n + n where 8n and 4n can be found easily by shifting (or addition). You can see how multiplying some odd numbers might get confusing. For slightly larger products (larger than 255) this method will be really confusing because you will have to start juggling the HiLo register pairs. There is another alternative.

Let us say you are multiplying 245 * 137

245 = $F5 = %11110101

137 = $89 = %10001001

You know how to multiply it it longhand. How about in binary? If you write it out in the same longhand method, it looks like this:

```
                  11110101
                  10001001
                  --------
                  11110101
                       Ø
                      Ø
             11110101
                  Ø
                 Ø
                Ø
       11110101
       ----------------
1000001100011101  = $831D
                  = 33565
```

Now the question is how to code this multiplication? It is relatively simple if you remember to get your shifts right.

```
*******************************************************************************
*
*
*       16 BIT MULTIPLICATION ADOPTED FROM LEVENTHAL 68K.ASM PROG PAGE 8-11
*
*       EXPECTS D1 = NUMBER1.W
*               D2 = NUMBER2.W
*
*       USES    DØ,D1,D2,D3
*     PRESERVES DØ,D2,D3
*
*       RETURNS DØ = 32 BIT PRODUCT
*
MULT_16         MOVEM.L DØ/D2/D3,-(A7)
                MOVEQ   #Ø,DØ           * CLEAR FOR PRODUCT
                MOVEQ   #16-1,D3        * bit COUNTER
*
LOOP            ADD.L   DØ,DØ           * SHIFT PRODUCT    LEFT 1 BIT
                ADD.W   D2,D2           * SHIFT MULTIPLIER LEFT 1 BIT
                BCC.S   STEP            * IF MULTIPLIER BIT =Ø:
                ADD.L   D1,DØ           * ELSE ADD MULTIPLICAND
STEP            DBRA    D3,LOOP
                MOVE.L  DØ,D1           * RETURN PROD IN D1
                MOVEM.L (A7)+,DØ/D2/D3
                RTS
*
*******************************************************************************
*
*
```

Notice that two 8 bit operands produce a 16 bit product and in general two n bit operands will produce a 2n bit product. Once you have the method down, you can see how to multiply two 32 bit numbers.

```
*
*******************************************************************************
*
*
*       32 BIT MULTIPLICATION AFTER LEVENTHAL 68K.ASM PROG PAGE 8-11
*
*       EXPECTS D1 = NUMBER1.L
*               D2 = NUMBER2.L
*
*       USES    DØ,D1,D2,D3
*     PRESERVES DØ,D3
*
*       RETURNS DØ:D1 = 64 BIT PROD
*
*     MSB                                   LSB
*     DØ                    D1
*    $1F - $1Ø:$ØF - ØØ ¦ $1F - $1Ø:$ØF - ØØ
*
```

```
MULT_32         MOVEM.L A0/D2-D5,-(A7)
                MOVEQ   #0,D5                   * CLEAR MS LWORD FOR PRODUCT
                MOVEQ   #0,D4                   * CLEAR LS LWORD FOR PRODUCT
                MOVEQ   #0,D0                   * CLEAR FOR ADDING CARRY
                MOVEQ   #32-1,D3                * BIT COUNTER
*
MLOOP           ADD.L   D4,D4                   * SHIFT PRODUCT    LEFT 1 BIT
                ADDX.L  D5,D5                   * NOW SHIFT MS LWORD
                ADD.L   D2,D2                   * SHIFT MULTIPLIER LEFT 1 BIT
                BCC.S   MSTEP                   * IF MULTIPLIER BIT =0:
                ADD.L   D1,D4                   * ELSE ADD MULTIPLICAND
                ADDX.L  D0,D5                   * ADD IN CARRY + 0
MSTEP           DBRA    D3,MLOOP
*
                MOVE.L  D5,D0                   * RETURN MS LWORD IN D0
                MOVE.L  D4,D1                   * RETURN LS LWORD IN D1
                MOVEM.L (A7)+,A0/D2-D5
                RTS
*
*******************************************************************************
```

I have been ignoring the most useful 'hardware' multiply instruction on the 68xxx, MULU (or MULS). As I mentioned above, this instruction uses only 16 bit operands to produce a 32 bit result. This is sufficient for most indexing uses, but rarely for arithmetic. eg. MULU #137,D0 will calculate the previous product in one instruction.

What if you want to calculate 1.2345E9 * 2.345E12 ? Then you are slowly and gently led down the path of n-bit multiplication. You can do 32 bit multiplication by utilizing the MULU instruction creatively.

```
*
*******************************************************************************
*
*
*       EXPECTS D1 = NUMBER1.L
*               D2 = NUMBER2.L
*
*         USES    D0-D6
*      PRESERVES D3-D6
*
*         RETURNS D1:D2 = 64 BIT PRODUCT
*
*          D1 = NUMBER1.L       a:b
*          D2 = NUMBER2.L       c:d
*      D1*D2 =  (a:b)*(c:d)
*            =  [(a * 2^16)+b]*[(c * 2^16)+d]
*            =  ac * 2 ^32 + (ad+bc) * 2 ^16 + bd
*
LONGMULT        MOVEM.L D3-D6,-(A7)
                MOVEQ   #0,D4                   * CLEAR FOR D4:D5 RESULT
                MOVEQ   #0,D5
                MOVE.L  D2,D3                   * GET C:D
                MOVE.L  D1,D0                   * GET A:B
                MULU    D0,D3                   * BD
                MOVE.L  D3,D5                   * STORE BD
                MOVE.L  D2,D3                   * GET C:D
                SWAP    D0                      * B:A
                MULU    D0,D3                   * AD
```

```
                SWAP    DØ                      * A:B
                SWAP    D2                      * D:C
                MULU    D2,DØ                   * BC
                ADD.L   D3,DØ                   * AD+BC
                BCC.S   NO_CARRØ                * NOTE 16 BIT OFFSET TO REGS
* IF CARRY: ADD $ØØØ1ØØØØ TO D4
                SWAP    D4
                ADDQ.W  #1,D4                   * ADD IN OFFSET CARRY
                SWAP    D4
NO_CARRØ        MOVE.L  DØ,D6                   * COPY AD+BC
                SWAP    D6                      * GET MS WORD
                LSL.L   #8,DØ                   * SHIFT TO HIGH WORD
                LSL.L   #8,DØ                   * SHIFT TO HIGH WORD
                ADD.L   DØ,D5                   * CALC LS LWORD
                ADDX.W  D6,D4                   * ADD X MS WORD
                BCC.S   NO_CARR1                * NOTE 16 BIT OFFSET TO REGS
                SWAP    D4
                ADDQ.W  #1,D4                   * ADD IN OFFSET CARRY
                SWAP    D4
NO_CARR1        SWAP    D1                      * B:A
                MULU    D1,D2                   * AC
                ADD.L   D2,D4                   * CALC MS LWORD
*
                MOVE.L  D5,D2                   * LS LWORD
                MOVE.L  D4,D1                   * MS LWORD
                MOVEM.L (A7)+,D3-D6
                MOVEQ   #Ø,DØ
                RTS
*
*******************************************************************************
```

For my Mandlebrot program, the multiplication I need to do is in Fixed Point arithmetic. All sorts of FP formats are possible. I have heard of people using 3 bits for the Integer and 29 bits for the fraction; and 4 bits for the integer and n bits for the fraction for n up to 1Ø24. One format I used because it was easy to code was 16 bits Integer and 16 bits Fraction. In this format, the multiplication routine came out like this:

```
*
*******************************************************************************
***
*
*  EXPECTS FIX#'s IN DØ,D1
*  RETURNS PRODUCT IN D1
*          NO ERROR FLAGS
*       FIXED POINT FORMAT: IIII^FFFF
*       WHERE: I = INTEGER PORTION
*              F = FRACTIONAL PORTION = INTEGER/65536
*
*       [IØ+FØ][I1+F1] = IØ*I1 + IØ*F1 + I1*FØ + FØ*F1
*
FIX_MULT        MOVEM.L D2-D5,-(A7)
                CLR.W   D5                      * DEFAULT SIGN + ie Ø's
                TST.L   DØ                      * SIGN?
                BEQ.S   PRODZERO
                BGE.S   ONE_POS                 * IF POSITIVE DO NOTHING
ONE_NEG         NEG.L   DØ                      * MAKE +
                TST.L   D1                      * SIGN?
                BEQ.S   PRODZERO
                BGE.S   RESULTNEG               * IF +: JUST SIGNAL RESULT SIGN
```

```
                NEG.L   D1                      * MAKE +
                BRA.S   FM_CONTØ                * IF -: RESULT WILL BE +
ONE_POS         TST.L   D1                      * SIGN?
                BEQ.S   PRODZERO
                BGE.S   FM_CONTØ
                NEG.L   D1                      * MAKE +
RESULTNEG       MOVEQ   #-1,D5                  * SIGN -  ie. 1's
*
FM_CONTØ        MOVE.L  DØ,D2
                MOVE.L  D1,D3
                SWAP    D2                      * GET IØ
                SWAP    D3                      * GET I1
                MULU    D3,D2                   * D2.L=I1 * IØ
                SWAP    D2                      * RETURN TO INTEGER POSITION
                MOVE.L  D2,D4                   * SAVE RESULT

                MULU    DØ,D3                   * D3.L=FØ * I1
                ADD.L   D3,D4                   * D4.L=I1*IØ + FØ*I1

                SWAP    DØ                      * GET IØ
                MOVE.W  D1,D3                   * GET F1                MULU
DØ,D3                   * D3.L=IØ * F1
                ADD.L   D3,D4                   * D4.L=I1*IØ + FØ*I1 + F1*IØ

                SWAP    DØ                      * GET FØ
                MULU    D1,DØ                   * DØ=FØ/64K * F1/64K
                MOVE.W  #Ø,DØ
                SWAP    DØ                      * DØ=FØ*F1/64K
                ADD.L   DØ,D4                   * D4.L=D4.L> + FØ*F1
                MOVE.L  D4,D1                   * GET RESULT
*
MULTSIGN        TST.W   D5
                BEQ.S   FM_EXIT
                NEG.L   D1
FM_EXIT
                MOVEM.L (A7)+,D2-D5
                RTS
PRODZERO        MOVEQ   #Ø,D1
                BRA.S   FM_EXIT
*
*****************************************************************************
*
*
```

I have yet to write the ultimate n-bit * n-bit Fixed Point multiplication routine, however the need is there & sooner or later the topic will arise. I wonder if there is anybody who has interfaced an MC68881/2 FPU to the QL?

>eof

32K N.V.M. SYSTEM REVISITED

Wilf Rigter designed the 32K N.V.M. board for a standard system which consists of a TS1000/ZX81 + 32K N.V.M. board + 16K TS1016 rampack. To work efficiently with the standard SINCLAIR rampack, Wilf separated the A14' line from the A14 line by means of resistor R2. Also, the A14' line is connected to the IC's through diode D3. For any system not using the SINCLAIR rampack join A14' to A14 by placing a jumper over R2 and lift out one end of diode D3. See schematic for placement of removable jumpers A and B if you need the flexibility.

If you run "HOTKEY" or "NEWROM" from a board which also serves as a system RAM, you cannot use the write protect switch. To write protect for example your "RAMDOS 1000" in the 0 to 8K region, power up in SINCLAIR ROM, switch in RAMDOS, bring your program into RAM, and switch back to SINCLAIR ROM for power down. In case you accidentally powered down in RAMDOS, here is how to restore it: (Let the board closest to the computer be 'A' and the board plugged into it 'B')

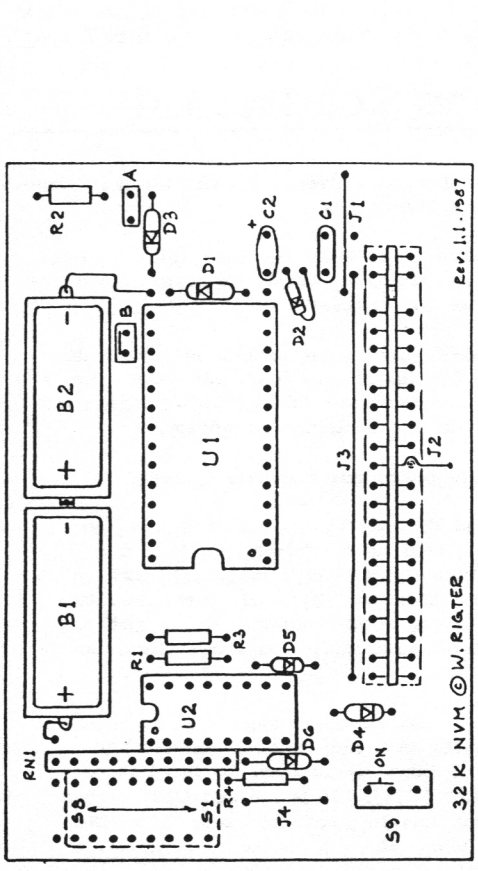
Board 'B': S8 off, write protect open

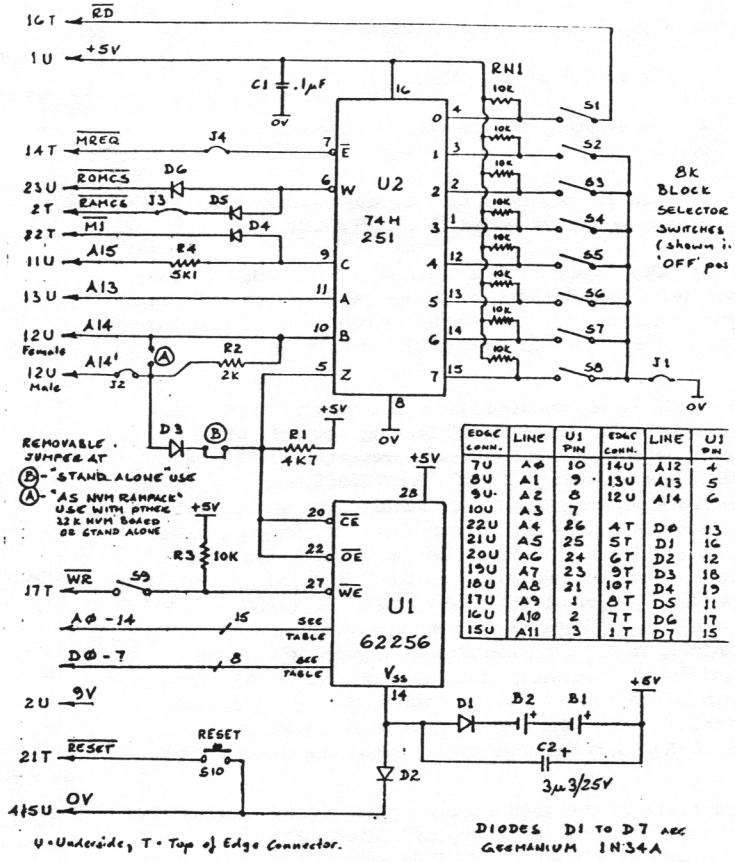
Board 'A': S8 on, S1 off

Load RAMDOS from tape, install HOTKEY or NEWROM. Close S8 and write protect on 'B', open S8 and close S1 on 'A'.

I am using my system with 'A' set for 0 to 32K and 'B' set for 32 to 64K, fully utilizing all addressable 64K. For me the small precaution needed for power down is a small price to pay for the benefit gained. I use 2 write protected 'B' boards filled with programs and utilities (only one at a time, of course).

Gerd Breunung





EDGE CONN.	LINE	U1 PIN	EDGE CONN.	LINE	U1 PIN
7U	A0	10	14U	A12	4
8U	A1	9	13U	A13	5
9U	A2	8	12U	A14	6
10U	A3	7			
22U	A4	26	4T	D0	13
21U	A5	25	5T	D1	14
20U	A6	24	6T	D2	15
19U	A7	23	9T	D3	18
18U	A8	21	10T	D4	19
17U	A9	1	8T	D5	11
16U	A10	2	7T	D6	17
15U	A11	3	1T	D7	15

32 k NVM SCHEMATIC

© 1987 Wilf Rigter

Rev. 1.1

Fig. 1

THE CASSETTE CONNECTION

Reprinted by permission from SyncWare News Vol2 #3

by Fred Nachbaur

In Volume 1, I ran a series on improving cassette reliability on the ZX81. Well, now that the TS2068 has arrived, that's all obsolete, right? Wrong! If anything, in my experience the 2068 needs even more "outside help" in getting reliable loads.

First, let's take a look at the load signal itself. As you may have noticed, the actual program is preceded by a header which consists of about five seconds (4032 cycles) of an 806.5 Hz. tone. This "sync pattern" is presumably used to set timing and compensate for variations in recorder speed. This is followed by:

1. The type of save (program, numeric array, string array or code)

2. A name up to 10 characters

3. The length of the file

4. The starting line number, variable array or address

5. Then (in the case of program saves) the offset to the variables (VARS-PROG).

After the header is a brief silence (though I use that term loosely), followed by another 5 second sync pattern and the actual program or data. (This info is from the tech manual.)

The data is represented quite differently from the ZX81/TS1000. Instead of having pulses of fixed width, with five such pulses representing a "0" and nine pulses being a "1", the TS2068 uses the more conventional approach of sending a short pulse (period=.48 ms) for zeros, and a longer pulse (period=.96 ms) for ones. This is, by the way, the same kind of system used by SDS, Z-XLR8 and other fast-load programs for the ZX81. Although it is about four times as fast as the standard ZX81 load routines, it is still considerably slower than the "fast-load" routines; the average rate of data transfer is about 1400 baud (bits per second) whereas SDS, for instance, runs at about 3500 baud and Z-XLR8 is variable up to about the same speed.

According to the tech manual, the SAVE signal is processed with a low-pass filter whose corner frequency is 2.5 KHz., which is very close to the output frequency when sending all zeroes. As a result, the signal is considerably "rounded" before it gets to the recorder. This may help in preventing overshoots and harmonic effects ("beating" with the bias oscillator in the recorder), but it tends to make the signal mushier and less sharply defined in pulse-width. As a side note, the manual also claims that the SAVE signal is between 0.15 and 0.67 V p-p; if this were so, your recorder would be blasted so badly you'd only get garbage. Perhaps this refers to the signal at the edge connector, or perhaps they meant 0.15 - 0.67 MILLIvolts (.00015-.00067 V).

The LOAD section is low-pass filtered at a much higher breakpoint (23 kHz.), so the effect this filter has on the load signal is negligible; it would not even significantly reduce any horizontal blanking pulses that might find their way to the tape. The sensitivity of the LOAD section is even less than the ZX81/TS1000, so a comparator or op-amp pre-conditioner is definitely recommended for this machine. Also, because of the extreme rounding of the save signal, volume setting is more critical; the range over which the pulse width is within acceptable limits is quite narrow, even with a pre-conditioner.

Now the Good News!

Now, the good news. There is an almost absurdly simple way of fixing this.

1. Remove the 7 screws that hold the two case halves together, gently separate them and unplug the keyboard.

2. Remove the three screws that hold the board to the lower case-half, one near the speaker, one near the "Timex 2000" logo and one next to the cartridge connector.

3. Lift the board out and turn it over.

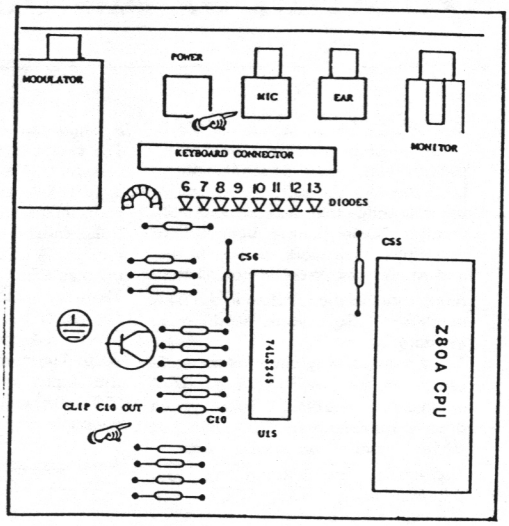
4. Shunted directly across the MIC jack, on the wiring side of the board, you'll find a small blue (on my machine) 120 pF. capacitor. Remove it, and throw it into your junk box. (Who knows, it might be useful for something.) This alone might do it.

5. There is another identical cap in parallel with this one, located about 2" to the left of the CPU and just to the left of an LS245 (U15). It is marked C10, though the legend may be hard to see; it's the smaller one, at the bottom of a row of components. Clip this one also.

6. Re-assemble the machine, and you're done.

Before I performed these "cap-ectomies" on my machine, I couldn't load tapes I'd saved on the TS2020 recorder, though other recorders worked sometimes. Now it works just fine, and is less sensitive to variations in playback level. What we've done is remove that low-pass filter and squared the save signal back up. It is unclear why C10 was included and downright obscure why another one was tacked on afterwards. Perhaps they were afraid of too much RFI and possibly flunking FCC testing.

If this results in too much noise on your machine, or if it didn't improve matters any, put one of the caps back in and try it again. It could be that you're getting harmonic "beating" with the recorder's bias oscillator. If you still have no joy, your problem is most likely with the load, rather than the save. Try a Winky board, VOTEM, the circuit in Volume 1 or other load signal pre-conditioner.



****

META MEDIA PRODUCTIONS 726 WEST 17TH VANCOUVER, BC CANADA V5Z 1T9

META MEDIA PRODUCTIONS ANNOUNCES Q_LINK 1.555

Now featuring: - UNARC & UNCRUNCH utilities for Msdos & CP/M files
- 7 & 8 Bit Data support - Full 7 & 8 Bit Parity support
- Expanded Editor functions - String Search, Goto, Append File
- Editable default path/filename - Toggle display of Control Chars
And Still featuring: - Supports Multitasking & Expanded Memory
- Dial,Redial - Integral Editor
- Xmodem & Ascii file transfer - 64 or 80 Columns
- ZOOM printing for speed - XON/XOFF handshaking
- Supports Multitasking & Expanded Memory
- Directory of any Device tells you File Type & Length
- Integral Editor for Capture Buffer, or Document Creation
Edit your session; mark a block, then Print, Save or Ascii Transfer it
Makes it easy to mark an interesting item & transfer it to another BBS
Store up to 40 Telephone Numbers, 20 Signons/Passwords per setup file
Edit phone numbers, BBS names & Signons painlessly to create setup file
Load another setup file for even more numbers & passwords. Complete
documentation. Extensive use of Menu/Quick modes for Novice & Expert.
Things are made easy with stored File Device, Printer Device & Baudrate
Configures to any modem. Set 8 seperate modem commands, parameters &
Messages; Dial, Immediate Redial, Reset. Supports all QL Baudrates
100% Machine Language for Speed! Works with a JSU, JM, MGUK ROMs
Includes the Utilities: Unsqueeze, Library, Filters, UNARC & UNCRUNCH

The Fine Print : Q_LINK 1.555 - US$ 29.95 + $3.00 shipping
Upgrade for registered owners - 10.00 + $3.00 shipping
Supplied on MDV or 5.25" disk [specify tpi]
META MEDIA PRODUCTIONS 726 WEST 17TH VANCOUVER, BC CANADA V5Z 1T9

QL RAM extension

Sinclair's QL has as standard a 128 K RAM, which sounds like a lot in comparison with most 64 K machines. Unfortunately, the software writers, in the knowledge that there is more than enough memory, have been rather wasteful in their work, so that at the end of the day, there is not all that much more in the QL than in the 64 K machines. So, you need more memory...

The accompanying circuit is an application of the TMS4500A as RAM extension for the 68008. This chip can drive a maximum of 128 K dynamic RAM and provides virtually everything: multiplexing of the address lines, $\overline{RAS}$, $\overline{CAS}$, and REFRESH.

The memory ICs are 64 K × 1 (128 or 256 refresh are both permitted) and have a speed of better than 150 ns. Since the QL uses a clock frequency of 7.5 MHz rather than the normal 8 MHz, such a RAM can run without wait cycles. An 8 MHz CPU that regularly has to carry out a wait cycle is appreciably slower than a 7.5 MHz type!

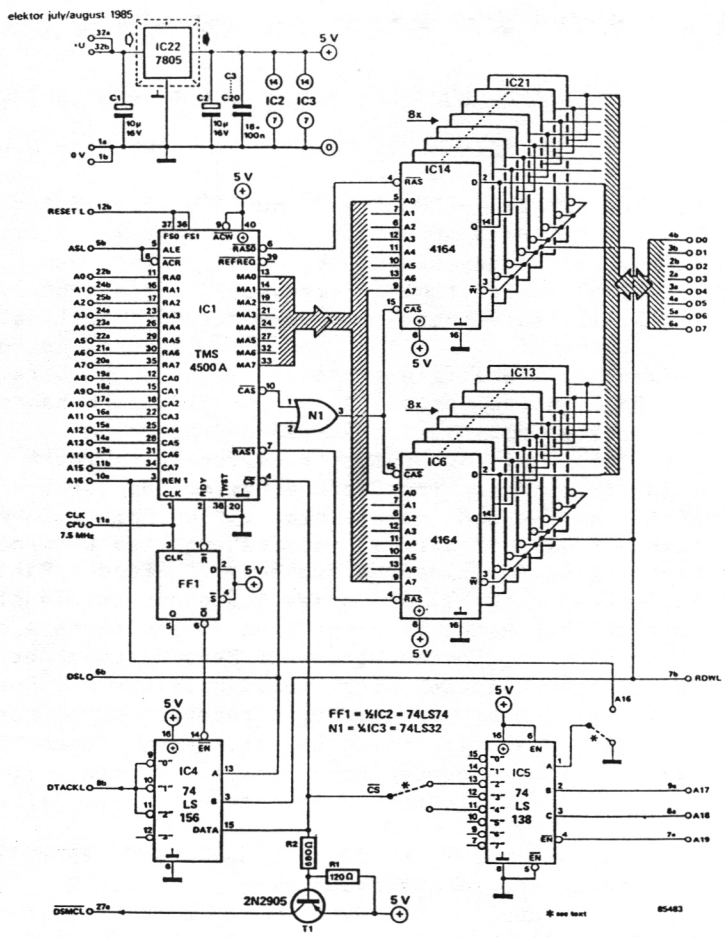
The 68000 family is provided with a data acknowledge input. As with other processors, the CPU places addresses and data onto the bus and indicates the validity with an address strobe and data strobe respectively. It continues to do so until the memory sends a $\overline{DTACK}$ signal. The present extension generates this signal with the aid of the LS156. Normally, this acknowledgment is given almost immediately, but it may happen that the 4500 is in the middle of a refresh. In that case, the CPU has to wait, which is arranged via the ready output (pin 2).

To prevent the QL waiting forever when an address is read that has no memory, the $\overline{DTACK}$ is generated internally: this must, however, be disabled for addresses where the RAM extension is located, and fortunately this can be done easily via $\overline{DSMC}$. By making this logic high as quickly as possible, the internal $\overline{DTACK}$ is cancelled.

If you cannot get the 2N2905 transistor, you may use a BS250, in which case resistor R_1 can be omitted and R_2 should be replaced by a wire link.

The circuit as shown is for the 128 K version. It is also possible to omit the eight RAMs connected to RAS1 and make a 64 K extension. Input A of the LS138 must then be connected to A_{16} and pin 11 instead of pin 13 must be used as $\overline{CS}$.

There is no 5 V supply available on the connector, but there is a 9 V line. This can be reduced to 5 V by a standard 7805. The current drawn depends on the types of RAM and will be 200...300 mA. It is important to decouple the supply lines properly: each RAM IC and the 4500 require a 100 n capacitor!



L O G I C A L O P E R A T O R S

A Course In Digital Electronics

By V. Lee

If you've never experimented with digital electronics but you've programmed in BASIC with such statements as,

```
10  IF A AND B = 3 THEN GOTO 50
20  IF C OR D = 7 THEN PRINT E
```

chances are you already know more about digital electronics than you suspect.

Unlike amplifiers, digital electronics is made up of logic switches called gates. They are concidered as either on or off, true or false, high or low. In the early days a person would build them out of relays but now they are available in integrated circuit form. There are two main families that are widely used, the 7400 series TTL and the 4000 series CMOS.

There are eight basic gate building blocks. A two input AND gate will produce a high output if both input A and input B is high. A two input OR gate will produce a high output if either input A or input B is high. An EXCLUSIVE OR gate will produce a high output if input A is different than input B. An INVERTER produces the opposite output condition from its input. Combining these gates with another INVERTER will produce four more new gates with the opposite conditions, the NAND gate, the NOR gate, the EXCLUSIVE NOR gate and the non-inverting BUFFER. When different gates are combined they can be made to perform elaborate tasks.

In 1847 George Boole developed a system called Boolean algebra. It showed how a logical condition could be translated into an equation. We can synthesize a logic circuit with ZX operators and have the computer perform all the calculations. This list which contain all the different input conditions is called the Truth Table. It allows us to analize the circuit without actually having to build it.

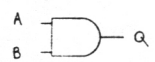
Example 1 shows the solution to the "Goat-Wolf-Corn" problem. A farmer empoys a hired hand that is not too bright. The hired hand is to keep the farmer's goat out of the corn barn while the door is open. There is also danger of a wolf that lurks nearby who will eat the goat. It is assumed that the situation is safe if the goat and the wolf are not visible and that the wolf does not eat corn. The farmer builds a circuit that contains three switches. All the hired hand need to do is to throw the switches when the situation occurs, door open, goat in sight, wolf in sight. The digital circuit will sound the alarm if a dangerous condition occurs which will alert the farmer. Does the circuit work? Try it and see.

The "Truth Table program" will calculate all the conditions for a logic circuit. It asks for the number of inputs and the equation. There are some examples available that may help you see how the program works. However there are two limitations. The program is limited to a maximum of six inputs numbered A to F although it can easily be modified to handle more. The other is that Boolean algebra will only work with combination logic, not with sequential logic. Sequential logic covers the area of flip flops, counters, shift registers etc. which relys on the previous conditions. It would require a much more sophisticated program and that may come in some future time.

If you're interested in learning more about digital electronics there are some very good books on the market. The favorites have always been the "TTL Cookbook" or the "Cmos Cookbook" by Don Lancaster. Other good ones include "Cmos/TTL, A User's Guide with Projects" by Joseph J. Carr. And the example for the "Goat-Wolf-Corn" problem is from "Digital Computer Circuits & Concepts" by Bill Deem, Kenneth Muchow and Anthony Zeppa.

EIGHT BASIC GATES

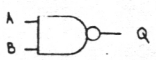
AND GATE



TTL 7408 CMOS 4081
INPUTS 2
ZX EXPRESSION A AND B

B	A	Q
0	0	0
0	1	0
1	0	0
1	1	1

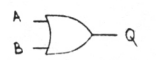
NAND GATE



TTL 7400 CMOS 4011
INPUTS 2
ZX EXPRESSION NOT (A AND B)

B	A	Q
0	0	1
0	1	1
1	0	1
1	1	0

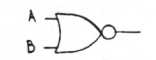
OR GATE



TTL 7432 CMOS 4071
INPUTS 2
ZX EXPRESSION A OR B

B	A	Q
0	0	0
0	1	1
1	0	1
1	1	1

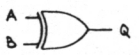
NOR GATE



TTL 7402 CMOS 4001
INPUTS 2
ZX EXPRESSION NOT (A OR B)

B	A	Q
0	0	1
0	1	0
1	0	0
1	1	0

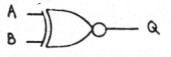
EXCLUSIVE OR GATE



TTL 7486 CMOS 4070
INPUTS 2
ZX EXPRESSION A<>B

B	A	Q
0	0	0
0	1	1
1	0	1
1	1	0

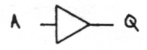
EXCLUSIVE NOR GATE



TTL 74266 CMOS 4077
INPUTS 2
ZX EXPRESSION A = B

B	A	Q
0	0	1
0	1	0
1	0	0
1	1	1

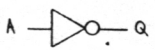
BUFFER



TTL 74365 CMOS 4050
INPUTS 1
ZX EXPRESSION A

A	Q
0	0
1	1

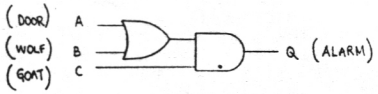
INVERTER



TTL 7404 CMOS 4049
INPUTS 1
ZX EXPRESSION NOT A

A	Q
0	1
1	0

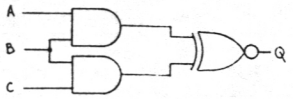
EXAMPLE 1



GOAT - WOLF - CORN PROBLEM

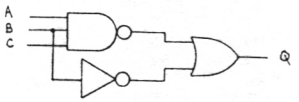
INPUTS 3
ZX EXPRESSION (A OR B) AND C

EXAMPLE 2



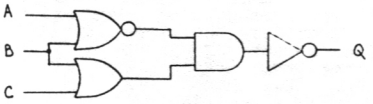
INPUTS 3
ZX EXPRESSION A AND B = B AND C

EXAMPLE 3



INPUTS 3
ZX EXPRESSION NOT (A AND B AND C) OR NOT B

EXAMPLE 4



INPUTS 3
ZX EXPRESSION NOT (NOT (A OR B) AND (B OR C))

```
  1 REM      THE TRUTH TABLE
                          BY V. LEE

THIS PROGRAM WILL PRODUCE A
TRUTH TABLE FROM A GIVEN BOOLEAN
EXPRESSION WRITTEN WITH COMPUTER
OPERATORS.

WHEN THE PROGRAM STOPS, ENTER

          Z  TO COPY
          C  TO CONTINUE

  5 GOTO 100
 10 LET A=Y
 15 RETURN
 20 LET B=Y
 25 RETURN
 30 LET C=Y
 35 RETURN
 40 LET D=Y
 45 RETURN
 50 LET E=Y
 55 RETURN
 60 LET F=Y
 65 RETURN
100 LET L$="F  E  D  C  B  A  "
110 LET U$="
"
120 LET S$="
"
130 PRINT AT 4,7;"THE TRUTH TAB
LE";AT 11,2;"ENTER THE NUMBER OF
 INPUTS";AT 13,11;"(MAX. 6)"
140 INPUT I
150 IF I>6 THEN GOTO 140
160 PRINT AT 11,1;"ENTER THE BO
OLEAN EXPRESSION";AT 13,11;"
"
170 INPUT E$
200 CLS
210 PRINT E$;AT 4,0;L$(19-3*I T
O );"  0";AT 5,0;U$( TO (I+1)*3)
220 FOR X=0 TO 2**I-1
230 LET N=X
240 FOR L=1 TO I
250 LET Y=(N-2*INT (N/2))
260 LET N=INT (N/2)
270 LET S$((I+1-L)*3-2)=STR$ Y
280 GOSUB L*10
290 NEXT L
300 LET S$((I+1)*3)=STR$ VAL E$
310 PRINT S$
320 NEXT X
330 STOP
340 GOTO 120
```


****

Larken Electronics' LKDOS

The Last and Best Hardware for Your 2068

by John Riley

A year ago I had made up my mind that I had maximized my 2068 system. G. Russell's Romswitch gave me access to the richness of British Spectrum software. The Aerco printer interface had allowed me to use a full-sized printer. A "recycled" Westridge modem hooked me up to the outside world. The Aerco FD-68 Disk Interface gave me mass storage, RGB output, and 64K of additional memory that nobody could give me any practical way to use. The only thing that I wanted and didn't have was disk storage for my Spectrum software, which Aerco promised but never delivered. So I resigned myself to cassette storage for the rest of my life, and always thought twice before acquiring much Spectrum software for this reason.

Then came the tantalizing rumor that Canadian hardware wizard Larry Kenny had developed a 2068 disk interface that was Spectrum compatible, had a Kempston joystick interface, extended Basic, and a pushbutton "snapshot" NMI save. The cost was some $80 less than the Aerco interface, but I was not thrilled with the idea of throwing my investment in the Aerco system out the window, nor did I relish the thought of having to convert my rather large disk library to a new DOS. So I stayed with what I had, while the Larken interface (LARry + KENny, get it?) gained a great deal of popularity among Canadian users, and made some inroads into the U.S. among those who had not already committed themselves to Aerco or Oliger.

During my stint as editor of this newsletter an even more interesting rumor appeared -- that Mr. Kenny was adapting the cartridge port component of his system to drive the Aerco interface with his own LKDOS. I promptly wrote to Mr. Kenny and did my best to egg him on in this project, assuring him that he would sell at least ONE such card -- to me! He wrote back and told me that a good bit of interest had been expressed in the idea, and he was forging ahead with it.

In late September 1987 I got an announcement that the LKDOS cartridge was ready for sale. Unfortunately this was right in the middle of my relocation from Maryland to Georgia, and it was the midst of December before I got around to ordering one. In mid-January, the package came! Although Mr. Kenny penned a short apology for the delay, I thought that this was a very reasonable turnaround time for him, especially in view of the delays that often accompany mail that crosses the U.S./Canadian border.

The package contained a ten-page manual on LKDOS, a three-page supplement for the Aerco loadable version, a disk full of demonstration software and utilities, and the cartridge itself. This latter is a hardware-crammed board that just barely fits in the cartridge port with enough room to close the cover. After making one small modification to the Aerco disk interface (details to follow), you plug in the Larken board and miracles begin to happen! The disk drive now works in both Spectrum and 2068 modes, the 64K onboard Aerco ram is available to function as a ramdisk (people who have the 256K version can have FOUR ramdisks!), and you have new windowing and graphics capabilities to play with. You can generate disks that are compatible with users of Oliger, Ramex, Aerco, or Larken disk systems, if they also have the Larken LKDOS cartridge. This virtually unifies the disk-based 2068 user world, as long as they have made the investment in LKDOS. As a further bonus, the Aerco eprom can be replaced with a Spectrum rom or Zebra Systems' OS-64 eprom, if the user wishes to get away from Aerco DOS entirely.

The modification to the Aerco board

Reprinted from the Mar/88 issue of the N/L of the Capital Area T/S Users Group

is a simple one, even to a certified member of the Hacker's Klutz Society like myself. Instructions in the manual supplement lead you to remove the plastic cover from the front of the interface. Be sure that you pry against the plastic and not against the circuit board, which is covered with copper foil tracery that is possible to damage! Then one is led to the fifteenth pin of the top row of the edge connector (counting from the left), where the NMI line is soldered directly in the form of a thin wire-wrap type of wire. This wire must be cut and an SPDT switch (I used Radio Shack part no. 275-613) spliced in. Actually, this is not quite as easy as it sounds, for said wire is only about an inch long. The best way to approach the task is to nip the existing wire where it solders onto pin 15, carefully strip the insulation off the end and extend it with another length of wire-wrap material. This kept me from having to solder directly onto an IC pin on the board, which makes me nervous! Then tapping into pin 15 on the back (feed-through) side of the board where there is more room to work, route the second wire to the front through one of the existing drill-holes on the board. The switch is then wired in, and the problem arises as to where to mount the switch. My kids use my 2068 a lot and it wouldn't do to just leave it hanging out of the circuit board. My solution was to build up the adhesive foam spacers that originally secured the front cover of the board, install the switch in a hole cut into the foam, and replace the protective front over the whole assembly. You can get a roll of adhesive foam stripping at any good hardware store. It is usually used for mounting light-weight pictures on the wall. You must cut off the bottom inch or so of the interface cover so that the edge connector can still fit into the back port of the 2068, but the back of the computer itself nicely fills this gap and protects the board from dust and children's fingers.

With the switch now in place you can selectively enable the NMI line and thus use AERCO DOS, or disable the NMI line, plug in thee Larken cartridge, and use LKDOS. Herin lies the only real complaint that I have with Mr. Kenny's cartridge, and it is perhaps a minor one. If the user wants to be able to use both the Aerco and Larken DOS, the board must continually be inserted and removed from the cartridge port. Alas, the board is so tightly packed with components that it is difficult to remove once it is in place! Someone with arthritis in their fingers might find it impossible to do so. I am going to deal with this problem by converting my most often-used programs over to LKDOS so that I can have access to them in either system. However, it would be much better if somebody would figure out a way to wire a switch that would cut the cartridge port in or out of the 2068 system without removing the cartridge! Does anybody out there know if this is possible?

Mr. Kenny's manual includes instructions for adding the NMI "snapshot save" pushbutton circuit to the Aerco board. I am going to attempt this in the near future and will be sure to send you a report on it. Until then I'll simply conclude by saying that every Aerco disk interface owner who has an interest in getting Spectrum material on disk or moving beyond the limitations of the Aerco system should seriously consider the Larken cartridge. At $65 for Spectrum compatibility, a ramdisk, a new and well-executed DOS, and several interesting and "do-able" optional hacks, it is a good investment indeed.

§§§§§§§§§§§§§

Silicon Mountain Computers
C-12, Mtn. Stn. Group Box
Nelson, BC V1L 2J3
Canada

USING THE PC8300 WITH 64K RAM PACKS

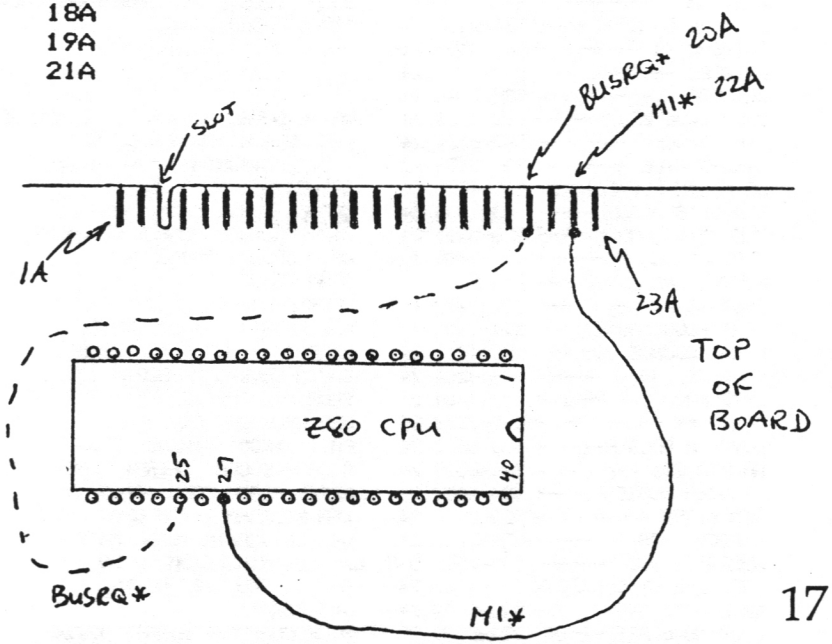
The PC8300 can easily be made to work with 64K RAM packs, as well as the 16K and 32K packs from Memotech. The reason it doesn't work presently, is that some of the CPU control lines were not brought out to the expansion connector. Some of these are unlikely to be useful; however, at least on (M1*) is absolutely VITAL to proper operation with large RAMs.

You can easily add this line. Simply run a wire from pin 27 of the CPU chip, to edge trace 22A. See the diagram below.

Some peripherals use the BUSRQ* line, so while you're at it you might as well add this one also. Connect CPU pin 25 to edge trace 20A.

The other missing lines are INT*, NMI*, HALT*, BUSAK*, WAIT*, and RESET*. Only the more exotic peripherals will need these, so casual users need not be concerned with these. However, if you wish to add these, the connections are as follows:

	CPU	EDGE CONNECTOR
INT*	16	11A
NMI*	17	12A
HALT*	18	13A
BUSAK*	23	18A
WAIT*	24	19A
RESET*	26	21A



BBS LISTING FOR THE LOWER MAINLAND
VERIFIED AND CURRENT AS AT MAY 5, 1988

BBS	PHONE	HOURS	COMMENTS	BAUD
1000 PLUS	597-1514	24	LEN BOSCOE OPUSNET	3/12
A & B SOUND BBS	321-5143	24	VARIOUS ATARI MICHTRON	3/12/24
A DIFFERENT CORNER	576-9602	24	DARK KNIGHT BLUE BD.	3
A WAYWARD SPARROW	984-6984	24	ANDY ANDERSON COMM.	3
ABACUS	272-4311	24	JOHN GYULASI X-MODEM	3/1200
AGORA BBS	463-4811	24	DAVID LOCKHEAD IBM FIDO	3/12
AIRSPACE SOCIETY	270-6475	24	DALE JACKAMAN RBBS	3/12
AMI HAVEN	584-1570	24	JOHN G. AMIGA BBS	3/12
APPLE PEELERS BBS	437-9222	--	MAVERICK APL PROTREE	3/12/24
ARCADIA	538-1074	24	WILD STRIKE COMM BLUE	3/12
ARCANE BIMMER	875-9788	24	ENCHANTER COMM BLUE BD.	3
B&B'S VIC20/C64	985-5042	24	THE COMMODORE MAN AABS	3
BASIC'LY	584-9811	20	BOB SATTI IBM OPUS	3/12
BEST #1	536-0024	24	COCO PBBS 4.92	3/12
BIG-BOARD	272-4644	24	JOHN GYULASI COMM.POLITICAL	3/12
BINARY STOCK EXCH.	266-1531	24	JOHN C. ATARI FOREM ST	3/12
BLOOM COUNTY BBS	263-3843	24	STEVE DALLAS COMM. BLUE BD.	3
BLUE HELL	879-8676	24	SYSOP: BEELZEBUB	3/12
BOTTOM LINE	583-1085	24	TED NIEWENHUIZEN APL.	3/12
BROADWAY BBS	435-9427	24	BRUCE KNIPE IBM TURB.	3/12
CASTLE ARRGH!	327-9494	24	YELLOWBEARD COMM. VISION	3
CATALYST	433-0214	24	BRYAN BEDFORD IBM OPUS	3/12
CEN-TA BBS	987-9388	24	D.I. ORIG.	3/12/24
CIRRUS	535-1382	24	MIKE MCCLOUD COMM.	3/12
CITYLINK**	222-2000	24	STEVE BARER	3/12/24
CLUB PARADISE	574-4776	24	WARLOCK COMM BLUE BD.	3
COMMODORE CC	271-1082	24	GLEN & DE'WAYNE COMM. VISION	3/12
COMPUDOME BBS	464-6716	24	MR. MICRO	3
COMPUSERVE**	738-5157	24		3
COMPUTER KITCHEN	538-3839	24	GROG! AMIGA BBS	3/12
COMSTAR	521-0886	24	MIKE MCLEOD IBM	3/12
COMM-ONLY	272-9222	24	JOHN GYULASI MAY/JUNE PW=C-ONLY	3/12
COUNTRY CORNERS	534-9154	24	PIPER COMM SCBBS	3/12
DATAPAC #1	689-8601	24		3
DATAPAC	662-7732	24	24	3
DATAPAC	687-7144	24	12	3
DEEP COVE BBS	929-6183	24	WAYNE DUVAL PC-BBS ALL COMPUTERS	3/12
DEEP SPACE	530-5134	24	THE GORGON COMM BLUE BD.	3
DIAL-A-FILE	736-3453	22	STEVE FAIRBAIRN IBM 12/24	3
DR ROOTH SEX CLINIC	936-4925	24	APPLE JACK IBM	3
DRAGON'S WORLD	583-1536	24	DRAGON LORD COMM AA	3/12
ELECTRIC COMPANY	273-5394	24	HIGH VOLTAGE APPLE AE ONLY!	3/12
FORTH BOARD	434-5886	14	JACK BROWN IBM OPUS	3/12
FANTASY PARADISE	263-2682	24	COMM. BLUE	3
FAST 80 #3	738-2773	24	DEBBIE COOPER IBM	3/12
FAST PLUS MASTER	594-7398	24	MEL PATRICK TRS80 BOARD	3/12
FIRST SOURCE BBS	253-4312	24	RENE ERTZINGER IBM OPUS	3/12/24
FROG HOLLOW	469-0264	24	DAVID BOWERMAN RCP/M	3/12/24
GATE FROM THE PAST	228-1481	24	SODA MAN APPLE	3
GENIE **	437-7313	24	USE HALF DUPLEX!	3/12
GUYS 'N GALS	435-6662	24	PATTY TAGO TANDY OPUS	3/12
HACKER BBS	589-0683	24	GEORGE SCURR ATARI BBS	3
HAUNTED HOUSE	581-6087	24	PETER FORD COMM. AABS	3/12
HEM BOARD	929-3776	24	KEN STERDAN IBM OPUS	3/12
HIDDEN PASSAGE	224-7385	24	VENICE VERMIN COMM AACG	3
HYPERION BBS	685-4578	6-5	CHARLES MIDDLETON FIDO	3/12
INDEPENDENT CONNEC'N	321-1366	24	AYLWIN SHU IBM VORTEX	3/12/24
LATE-NITE	278-3569	24	THE SPARK	3/12
LOOKING GLASS	321-5930	??	RICK GUNNYON ATARI 12/24	3

MAGNETIC VISIONS ---	926-7192	24	POOBAH REAL-TIME CHAT	3
MIND LINK** -------	533-2312	24	REITER/ALLEN	3/12/24
MIND MELD BBS ------	438-7885	24	WEYMAN & JASON FIDO	3/12
MOONSTAR ----------	434-4945	10-10	S.J. SMYTHE IBM RBBS 12/24	3
MULTITECH BBS ------	733-1383	24	MICROPROFESSOR PCBOARD 10	3/12/24
NETWORK XXIII ------	438-5235	24	FIRE FOXCOMM AA	3/12
NORTHERN LIGHTS ----	588-0789	23	AARON WASS OPUS	3/12/24
ONE ON ONE BBS -----	590-1185	24	CORY MUZYKA VORTEX 2.	3
ONEIRO'S ORACLE ----	294-3897	24	ONEIRO OMM. VISION	3
OUT-LAW BBS -------	591-3472	22	DARYL & TYLER OPUS	3/12/24
PUBBS -------------	526-3389	24	GAVIN TI BOARD	3/12
PACCOM1 -----------	666-2981	24	JIM DEAN	3/12
PACIFIC SYSTEMS GRP.	228-9706	23	TED POWELL FIDO	3/12
PAPIER MACHE CAFE --	530-2357	24	THE MATRE DE COMM AABS	3/12
PEEP HOLE ---------	984-4079	24	SIX INCHES RCP/M PRIV.	3/12
PHANTOM BBS -------	939-4857	24	DANIEL CARRERAS AMIGA	3/12
QUESTOR PROJECT ----	681-0670	18	STEVE P. IBM OPUS	3/12
REALM OF SHADOWS I -	521-0006	24	NIGHTSHADOW COMM.	3
REALM OF THE KNIGHTS	946-8530	24	BLACK KNIGHT OPUS	3/12
REVELATION --------	929-1615	24	APPLE	12-96
RICHMOND LIBRARY ---	276-2278	??	LIBRARY HOURS 7-1-0	3
S.F.U./MTS -------	294-4180	24	INFO AT 291-32	3
SAGA BBS ----------	688-8536	24	VIKING ATARI BBS	3
SALVAGE EXCHANGE ---	591-9633	24	THE SCAVENGER COMM	3/12
SAM OBEN ----------	879-9871	24	SAM OBEN ATARI	3/12/24
SHADOW WARP BBS ----	270-7858	24	SHADOW KEEPER COMM.	3
SHORTCIRCUIT ------	594-4615	24	PENGO COMM AABBS	3/12
SILVER BULLET BBS --	873-3640	24	SILVER BULLET ATARI	3/12
SMOKEY MOUNTAIN ----	462-8753	24	SUSAN CARMACK RBBS	3/12/24
SNAKE PIT ---------	298-3530	24	COBRA COMM.	3
SOTA BLUE PLUS -----	688-5061	24	ANN JACKSON COMM BLUE BOARD	3/12
SPEAKEASY BBS ------	435-7699	24	TRIPLX RBBS 12/24	3
SPECTRUM BOARD -----	738-7606	24	BRIAN SIMPSON OPUS 12/24	3
STOCK CHAT ---------	254-7232	24	RIC CHISTE ATARI	3
SUNSHINE BBS ------	943-1612	24	BOB COTTER APPLE	3/12/24
T & P VANTARI -----	435-1727	24	DEVON SHEPPARD ATARI	3/12
T.R.A.C.E. --------	272-5888	24	BAD SECTORS ATARI 12	3
TERMINAL CITY ------	731-6966	24	DR. BENWAY COMM.	3/12
THE APEX ----------	685-8765	24	MASKED MAGICIAN APPLE	3/12
THE BEAST BOARD ----	585-7391	24	K. HERCUS OPUS (PETS)	3/12
THE CRYSTAL TALISMAN	421-3282	24	LONNIE W.V. NOCHANGE	3/12
THE DUNGEON --------	327-0848	24	WARLOCK COMM.	3
THE ELITE FORCE ----	597-2607	24	MAD SYSOP COMM.	3/12
THE FUNNY FARM -----	929-8812	24	MYRDDIN & ALKERION COMM.	3
THE PEACE MARCH ----	261-4495	24	KARATE KID COMM.	3
THE TOY (D+D) ------	946-7445	24	MR. MISTER APPLE GBBS2	3
TIME WARP ---------	943-2077	24	D. MCDONALD TI99-4A	3/12
TYMNET** ----------	683-7620	24	U.S. PACKET SWITCHING NETWORK	3
UBC LINE ----------	228-9851	24		3
UBC NET** ---------	228-1401	24	FACULTY/STUDENTS ONLY	3/12
UBC NIMNET ** ------	228-5011	24		3
UNDERSIDE BBS ------	939-2081	24	STRIDER OPUS	3/12
USER'S CHOICE ------	530-4722	24	UNCLE CLEM COMM	3
VANC'R PC USERS BBS	434-3434	24	IBM COLLIE 12/24	3
VANCOUVER FOGLIGHT -	271-5934	24	JAY SIEGEL RCPM/2	3/12
VERMILLION CROSSING	986-6529	24	THE TIME BANDIT COCO	3
ZILCO DELIGHTS	524-4824	24	SYSOP: THE OMINOUS ONE	3/12

ALL THE BBS'S ON THIS LIST WERE VERIFIED AND WERE ANSWERING THEIR LINES IN THE WEEK OF MAY 6, 1988.

Reprinted from the May/88 issue of the Nite-Time News - the N/L of the Chicago Area T/S Users Group

TELECOMMUNICATIONS ON THE QL

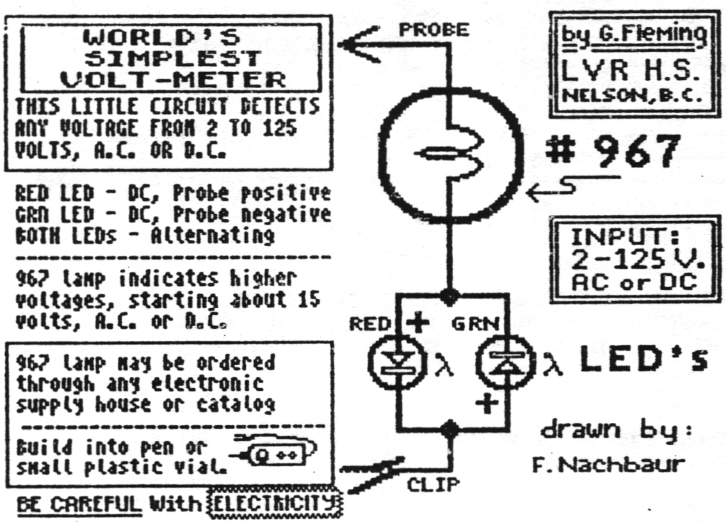
By Michael Mitchell
(Reprinted from QZX)

The Ql for all its power and versitility is limited in telecommunicating at 300 baud without a hardware link between the serial port and the modem. The following program which appeared in the Data Expansion or the Dallas/Fort Worth group will get you on line to a TNC or a telephone at 300 (or even 1200) baud.

```
  1 REMark         300/1200 baud terminal
  2 REMark                MIKE_TERM
  3 REMark               Developed by
  4 REMark             Michael Mitchell
  5 REMark           attribution requested
  6 REMark          Sinclair/Timex User Group
  7 REMark          Boston Computer Society
100 WINDOW 465,204,6,0
105 BORDER 1,0,7: LB=127
110 CLS: POKE 163976,255: CLS#0
120  PRINT#0,,"Choose Baud:   H=1200,  L=300
130 IF INKEY$(-1)=="h"THEN BAUD 1200: ELSE BAUD 300
140 CLS#0: OPEN#5,SER2i
150     PRINT#0,,"F5=QUIT","ALT^C=^C","^G=BELL"
160 REPEAT TERM
170 TERMINAL: PRINT#0,,"Exit?(y/n)"
180  IF INKEY$(-1)=="Y" THEN EXIT term
190 END REPterm
200 CLOSE#5
205 PRINT#0,,"Re-run/Quit? (r/q)"
210  IF INKEY$(-1)=="R" THEN RUN: ELSE STOP
220 DEFine PROCedure terminal
230  REPeat key_loop
240   a=CODE(INKEY$(#5,0))&&LB
250    IF a=13 THEN PRINT
260    IF a=7 THEN BEEP 2000,15
270    IF>31 THEN PRINT CHR$(a)
280   b=CODE(INKEY$)
290    IF b=248 THEN EXIT key_loop
300    IF b=255 THEN b=0
310    IF b=0 THEN NEXT key_loop
320    IF b=10 THEN b=13
330    IF b=208 THEN b=7
340   PRINT#5, CHR$(b);
350  END REPeat key_loop
360 RETurn
370 END DEFine terminal
```

Save as MIKE_TERM_bas

* *



The Trump Card

by David Hoshor

Reprinted from the Jan/Feb issue of The Ramtop - the N/L of the Greater Cleveland Sinclair User Group

The Trump Card is simply the most useful single addition you can get for the Sinclair QL. It combines a disk interface, 768 kilobyte RAM expansion, Toolkit II for SuperBASIC extensions, a screen dump, static and dynamic ramdisks and a dynamic printer buffer. When you consider that even the lowest cost 512 Kb RAM expansion and a disk interface will cost at least $250, and Toolkit II costs $70, the Trump Card is a bargain at $309.95. The entire unit only sticks out about three inches from the expansion port side of the QL, so it's compact. I got mine from Curry Computer in Glendale, Arizona.

The disk interface is pretty standard fare. It will allow you to store information on any combination of forty and eighty track drives, single sided or double sided, 5 1/4" or 3 1/2". The interface is fairly intelligent in that it will pick up on what format the disk was created in automatically. For example, if you have a 80 track, 5 1/4" double sided drive, the interface will automatically read from a 40 track, single sided disk. Of course, you can't make a single sided drive read from a double sided disk, or put a 3 1/2" floppy in a 5 1/4" drive since the interface can't change the limitations of your disk drives. But if you have an 80 track 5 1/4" double sided drive, you'll be able to read any disk with the exception of 3 1/2" disks. The interface is only able to support two floppy disk drives. That's its only shortcoming as far as I'm concerned.

The 768 Kb RAM expansion maxes out the QL. Added to the 128 Kb that are built into the QL, you have a whopping 896 Kb of RAM. (While the 68008 CPU can address 1 Mb of RAM, 128 Kb of memory in the QL are reserved for device addresses and ROMs.) You'll have room to run any program that has ever been written for the QL - probably several copies of the program. This can be very helpful on a multitasking machine like the QL. You'll also have plenty of room for ramdisks too.

The ramdisking capabilities of the Trump Card are really neat. The ramdisks come in two varieties, static and dynamic. The difference is that the static ramdisk has to be formatted, almost like a microdrive cartridge, and the dynamic ramdisk can be created by just using the device name. For example, to create a static ramdisk with two hundred, 512 byte blocks of space, just enter:

```
FORMAT RAM1_200
```

A 100 Kb section of memory will be set aside for use by the machine to use just like a microdrive or floppy disk, but will be located in fast RAM. To create a file in dynamic ram, it's just as simple as using the device's name. To copy a file to ramdisk ram3_ just enter:

```
COPY dev_filename to RAM3_filename
```

A bit of memory will be set aside in RAM as ram3_filename. It will grow only large enough to hold the file(s) that have been send to ram3_. The difference between the static and the dynamic ramdisks is that the dynamic ramdisks are more likely to fragment the memory of the QL. If the memory becomes sufficiently fragmented, it can degrade the performance of the machine because the operating system will have to scrounge around to find blocks of memory. Up to eight ramdisks can be used with the Trump Card. One other neat feature of the ramdisk is that it can make a very fast image of everything on a microdrive. It will copy everything on the microdrive in just about fifteen seconds.

One of my favorite features of the Trump Card is its printer buffer. It's a snap to use and can really save time. In its simplest form, you just use the device name "PRT" instead of "SER". What happens is that whatever you are sending to the printer is sent in its entirety to a buffer in RAM. The QL then sends bytes to the printer as a background task. The advantage to this is that you can return almost immediately to an editing session if you send a file to "PRT" from The Editor or Quill. If you send a file to "SER" from an editor, you must wait until the printer has finished printing before you can continue editing. Several files can be sent to the printer buffer and they will be sent in the proper order. There is a method of changing the device that you want to have the "PRT" device use. Unless you have a parallel interface that doesn't use the serial port, it's hardly likely that you'll change it.

The printer dump is pretty good. It supports about any Epson dot matrix printer plus a few other printers including the Brother HR4, Olivetti JP101, Seikosha GP-100A, GP-250X, and GP-700A, and the Canon PJ1080A. You can specify what section of the screen to copy, and there is a provision for using a "hotkey" to copy any screen from any program that you might be running. There's also documentation on how to open a printer device and get a printer dump from programs written in nearly any computer language - not just SuperBASIC. I've tried it from "C" and it does work.

Toolkit II is a collection of SuperBASIC extensions. Some of them are much more useful than others, but on the whole, they are very worthwhile. To me, some of Toolkit's most useful commands are "ed", a screen based SuperBASIC editor; "wcopy", a file copier that allows the use of wildcards; "wstat", a wildcard directory command that also provides the file size and last update time of files; "splf", another printer buffer or rather "spooler" that works great when used with "prt"; and an assortment of job control commands that allow the user to monitor, change the status of, and remove jobs that are the in the QL. Toolkit II allows the use of default directories, has clock features including alarm clock(s), has improved SuperBASIC error handling, permits the user to define "hotkeys", altkey/single key combinations to enter large strings or commonly used commands, and has an easy way to repeat of the last command typed. Some of the more obscure commands deal with networking, direct unbuffered access to the various i/o devices, and memory management commands. All in all, its a pretty impressive array of extensions to SuperBASIC. Once you start using the Toolkit II commands, you'll never want to use a machine without them. An additional benefit of having Toolkit II fitted on the Trump Card is that it leaves the ROM cartridge slot on the back of the QL free for other ROMs.

When you consider the features that the Trump Card offers, I feel that it is an unbeatable combination.

Silicon Mountain Computers
C-12, Mtn. Stn. Group Box
Nelson, BC V1L 2J3
Canada

UPLOADING BIG MEMOTEXT FILES

This article was inspired by a customer who had BIG (around 44K) Memotext files created with V1, and needed to upload them to a mainframe. I decided to write it up in an info-sheet form, in case anyone else had the same requirement.

On first look, it would appear that the best way to proceed would be to split the large textfile into several smaller ones, which could then be uploaded as described in the ZX-TERM*80 AND MEMOTEXT article. However, on closer examination one comes to the realization that there simply isn't enough room to maneuver data around, when the data is 44K long and the whole system has only 64K available. Though it might be possible, using "dummy" files which are imported to using a binary tape-save program like Z-XLR8, the convolutions required would drive the most stable individual over the brink.

So if one computer can't cut the mustard, how about using two? (I've yet to meet a devoted ZX/TS user who only owns one machine.) "Computer #1" would have 64K, and would house a slightly modified version of Memotext along with the "mega-textfile." The modification to Memotext would be a custom "Printer driver" that simply shoots the data out through the modem, instead of to the printer. "Computer 2" would be running ZX-TERM*80, and would be used to capture the data transmitted by computer 1. Once captured, the data could be uploaded immediately, or saved to tape or disk, for later uploading to the remote system. This computer could be running as little as 16K, plus the 8K static RAM in 8-16K. About three pages of text could thus be "cross-loaded" at a time. With more memory (32-64K), up to about 5 pages could be absorbed in each "go."

"Computer 2" could be virtually any other computer system. For instance, if you have a friend in the neighborhood with a modem and a reasonable terminal program (i.e. capable of up/downloading), you could recruit his help. However, having full and immediate control is preferred, so if at all possible arrange for "Computer 2" to be within your work space.

THE HARDWARE

I tested this scheme using two computers (a 64K TS1500 and a 32K TS1000 with SCRAM board). Both were connected to a Westridge modem. The two modems were directly connected together. In other words, just plug the cable from one modem (doesn't matter which) into the jack on the other modem. Don't connect the other cable to the phone line! You don't even have to plug a phone into the empty jack!

That's all the hardware that is required.

THE SOFTWARE

For software, you will need Memotext V1 (Computer 1) and ZX-TERM*80 (preferably) or Mini-Xmodem (at the very least) for Computer 2.

Memotext will require a modification to the printer driver. If you have V1C-2.03 or above (supporting a variety of interfaces), facility is provided for entering your custom driver code. If you have the older V1 specifically for the Memotech interfaces, you will have to patch in the new code into the "RS232" driver (how appropriate!). This article will assume V1C-2.05 (the most recent version), but will point out differences in the other versions.

The code that we will add consists of three routines. The first is just "send character in A to the modem," and replaces the printer driver. The other two are used to initialize (turn on) and shut off the modem. The routines, and their run-time addresses, hexcode, and decimal values is given below:

```
ADDR HEXCODE NAME MNEMONIC    DECIMAL    REMARKS
-------------------------------------------------------

27C5 F5      DRVR PUSH AF     245        ;save character
     DB77         IN A,77     219,119    ;ready to send?
     CB47         BIT 0,A     203,71
     28FA         JR Z 27C6   40,250     ;no? then loop back.
     F1           POP AF      241        ;retrieve character
     D373         OUT (73),A  211,115    ;ship it out.
     C9           RET         201        ;return to Memotext

27D0 21E127  INIT LD HL,INVL  33,225,39  ;start of table
     0606         LD B,06     6,6        ;6 values
     7E           LD A,(HL)   126        ;get value to send
     D377         OUT (77),A  211,119    ;send to ctrl port
     23           INC HL      35         ;next entry
     10FA         DJNZ 27D5   16,250     ;loop until done
     C9           RET         201        ;return to BASIC.

27DC 3E15    OFF* LD A,15     62,21      ;15 to control port
     D377         OUT (77),A  211,119    ;turns off modem
     C9           RET         201        ;return

27E1 00      INVL DEFB        0          ;3 0's followed by
     00           DEFB        0
     00           DEFB        0
     40           DEFB        64         ;40h
     4F           DEFB        79         ;mode word for 8/N/1
     37           DEFB        55         ;turn on modem
```

If you have V1C, answer "6" ("Other Centronics") for your interface type. Then enter the decimal values shown above. If you have 2.04 or previous, you'll get "CODE TOO LONG" with an error stop when entering the last number. Don't worry about it; GOTO 610 gets you going again.

If you have the older V1M versions, or perhaps a customized version for the Byte-Back serial or whatever, arrange to poke these values in at address 18503 through 18536. Also POKE 18502 with 0. With V1M, answer "R" when asked "RS232/Centronics?" when subsequently running. With V1C you just have to press ENTER, since this prompt was removed.

USING THE SYSTEM

Now that you've got Memotext suitably modified so that it operates as a "ASCII uploader," here's how you would go about dividing up your massive files into manageable chunks.

Load your big text file into the modified Memotext as usual. Load ZX-TERM*80 or other terminal into computer 2, and relocate as appropriate to give you the maximum file space.

Initialize your DATA REM, but don't open your capture buffer (save toggle) just yet. Turn on modem 2.

When the text file has loaded, QIT to BASIC, and enter:
RAND USR 10192 (turns on modem 1)
RAND USR 13787 (returns to Memotext).

Now PTF. Answer Start page No. and Justify Y/N as desired. Answer Single-sheet mode. Before pressing P to print the first page, open your capture buffer. The first page will now be transferred.

Repeat with as many pages as you have room for. If you get "Buffer full", you're best off to restart. It won't take you long to discover how many pages you safely have room for.

When the last page in this "go" has been transferred, close your capture buffer (save toggle off). Quit ZX-TERM*80, answer SAVE YES, and save to tape. When done, re-enter ZX-TERM*80, turn modem 2 back on, re-initialize, and continue "printing."

When all done, turn off modem 1 by quitting to BASIC, and entering RAND USR 10204.

Though this may be time-consuming, it's pretty "mechanical" compared to some of the exotic tomfoolery with Hot-Z, Z-XLR8, etc. that would be required otherwise.

Fred Nachbaur

§§§§§§§§§§§§§§§§§§§§

PRINT OR LPRINT

By James F. Brezina

Each new book I have bought on the TS 2068 has taught me quite a bit about programming on the computer. The things I have learned lately on the keyword PRINT are quite interesting. The latest book I purchased, "Introduction to 2068 Machine Language" by Dr. Lloyd Dreger, explained quite a bit about it.

Many times I have entered programs with the command "PRINT#0;" or the command "PRINT #1;". I found that the command would cause whatever followed it (a string or numbers) to be printed to the bottom two lines on the screen. However, in order for that information to remain on the screen, one has to provide some means to prevent an error statement or INPUT from appearing there. That can be done by a PAUSE or by following it with along FOR - NEXT loop. All the PRINT # commands are to befollowed by a semi-colon.

Dr. Dreger's book informs me that "PRINT #2;" will print to the upper screen which is the same thing that PRINT also does. The next PRINT command "PRINT #3;" will send the printing to the printer. This will be either the 2040 printer or a full size printer as long as you have the printer driver loaded and initialized.

Is there a PRINT #4;? Yes, I have found it used by the "ZTALKER". It is the means by which words are entered to make the "ZTALKER" talk. However, some words do not sound right if spelled normally, so you might have to misspell them to get the"ZTALKER" to sound right.

I have not seen anything about using anything above #4 in these PRINT statements in the above manner. I have seen the mused in another manner which I will explain later on.

An interesting thing about this PRINT #3 setup is that, you can also enter LIST #3 and it will LIST on the printer. Another thing you can do is with the LPRINT and LLIST commands. LPRINT#2 and LLIST #2 will go to the screen instead of the printer.

A number of years ago, I saw an article on one of the uses of the OPEN # command. This was originally intended for use with disks, however, it can also be used for printing without a disk system. The manner in which it was used was to enter "OPEN #2". The 2068 will not let you enter "OPEN #2" alone but it will let you enter "CLOSE #2" by itself. To enter "OPEN #2" you must follow it with a comma (the comma is the only punctuation mark that works) and one of the following letters in quotation marks:

"S" for the upper part of the screen.

"K" for the lower part of the screen (with something like PAUSE to keep the print on the screen)

"P" for printing to the printer (any kind as in "PRINT #3"

This will cause anything in a PRINT statement to go to where the letter indicates. The most usefull way of entering this commandis, "OPEN #2,"P". After entering this command (whether immediate mode or in a program, everything in the program that is in a PRINT statement will go to the printer. The simplest way of redirecting the print to the screen is to enter "CLOSE #2". I have seen one article that said to enter "OPEN #2,"S", but, that to me is a waste of keystrokes and it still leaves the channel open.

Reprinted from the May/87 issue of Nite-Time News, the N/L of the Chicago Area T/S Users Group

I have found that the only channel that works that way is channel 2. You can use any one of the other 15 channels to send print statements to the printer, but, you must follow them with the command: "PRINT #(channel you are using);" followed by what you want printed. An example of this is as follows:

```
10 OPEN #5,"P"
20 PRINT #5;"Mary had a little lamb"
30 CLOSE #5
```

A while back I found a little program (I believe it was in TS HORIZONS) that works like a simple word processor. The original program was written as follows:

```
10 INPUT AT 21,0; AT 0,0; LINE A$
20 LPRINT A$
30 GO TO 10
```

What happens with this program when you run it, is a cursor appears on the top of the screen. As you enter letters they are printed to the top of the screen and the cursor moves ahead of the letters. The entered string does not have quotation marks. Almost everything works as normal except the down arrow. It is the BREAK key for this program. You can even use the CAPS LOCK for this program. You can enter GRAPHICS. When you key the ENTER key, what is on the screen is printed on the printer. The screen would then be erased. Of course, a full sized printer will not print the GRAPHICS. You can also use the ENTER key for a LINEFEED. For a full sized printer, you will have to have your printer driver loaded in and initialized.

I tried an alteration on the program by changing the 0,0 in line 10 to 1,0. Then I added a line 5 to print the numbers 1 through 0 all the way across the screen. I found that this line would remain on the screen at all times while the rest of the text would be erased with ENTER to print to the printer. I also found that corrections could be made to the text. I also tried putting a semi-colon after LPRINT A$. This had a drawback as one had to add spaces to fill the printers buffer or the entire text would not be printed out.

In the September issue of Time Designs Magazine, one writer asked if there was a way to get the 2068 to print direct to the printer without using a monitor. Tim Woods answered that he knew of no way of doing this. The next issue contained quite a few letters in answer to that question, but, none of them really gave an answer to do what the writer wanted. One of the answers gave me the following idea, but it still does not do what the writer wanted.

```
5 POKE 23692,2
10 LET A$ = INKEY$: PRINT A$;: LPRINT A$;
15 PAUSE 20
20 GO TO 5
```

The POKE 23692,2 makes the text on the screen scroll up when the screen fills instead of breaking out. The semi-colons after the A$ keeps the printing on one line, otherwise, there would only be one letter to a line. The PAUSE is necessary, as without it you would not be able to get your finger off a key fast enough so it wouldn't repeat. What happens is that the printer will print out a line of text when the printer's buffer is full or when you key ENTER. This program has a number of disadvantages. There is no cursor on the screen. You cannot

delete screen letters with the 0 key. You can move the unseen cursor with the arrow keys and correct words on the screen, but, you cannot change what is in the printer's buffer. The result is that your mistakes are still printed on the printer. You can still break out of the program with the CAPS SHIFT & BREAK keys. CAPS LOCK cannot be used.

I tried a number of ways to make a cursor appear in the text on this program. I had no luck. Maybe one of you might find a way.

****

TS/1000 VERIFY

by DAVID NOWOTNIK

With only 8K of ROM in the TS/1000 it's little wonder that it hasn't a verify command. This little program will take care of that omission.

This routine is based on the LOAD routine in ROM. The change is when a byte is read off the tape. Instead of putting the byte into the approprite place in RAM, it is compared with the current byte at that address. If their is not a match, then the routine exits with an error message (R/0).

If all bytes match and the verification was successful, with no mis-match, than the 0/0 message will be returned at the end of the routine.

First enter this machine code loading routing, with 135 spaces or characters in the REM statment.

```
 10 REM .....(135 SPACES).....
 20 LET X=16514
 30 INPUT A$
 40 IF A$="S" THEN STOP
 50 LET J=16*(CODE A$-28)+CODE
A$(2)-28
 60 IF PEEK X=27 THEN POKE X,J
 70 LET X=X+1
 80 GO TO 30
```

Next RUN the program and enter these HEX digits.

```
CD 23 0F 37 11 00 00 CB 12 CB
0A CD 10 7C 18 FB 0E 01 06 00
3E 7F DB FE D3 FF 1F 30 49 17
17 38 28 10 F1 F1 BA D2 7A 7C
62 6B CD 10 7C CB 7A 79 20 03
BE 20 D6 23 17 30 F1 FD 34 15
21 09 40 50 CD 10 7C 00 CD 6C
7C 18 F6 D5 1E 94 06 1A 1D DB
FE 17 CB 7B 7B 38 F5 10 F5 D1
20 04 FE 56 30 B2 3F CB 11 30
AD C9 7A A7 28 BB CF 0C EB 21
7C 40 37 ED 52 30 06 1A B9 28
02 CF 1A 13 2A 14 40 37 ED 52
EB D0
```

Now the routine is in the long REM statment, and you can delete lines 20 through 80. DO NOT delete line 10. After these lines are deleted, add the next lines, 20 through 70. Then SAVE this before you go further, it is your VERIFY program.

```
20 LET X=16514
30 FOR I=31744 TO 31878
40 POKE I,PEEK X
50 LET X=X+1
60 NEXT I
70 NEW
```

To use the VERIFY routine, it must be loaded into your TS/1000, above RAM-TOP, before any other program. First lower RAM-TOP with these three direct commands....

```
POKE 16388,123 -- ENTER
POKE 16389,255 -- ENTER
NEW -- ENTER
```

Now you may LOAD and RUN your VERIFY routine.

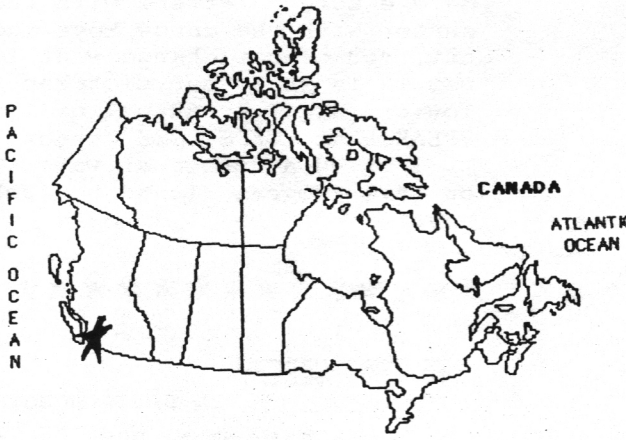
You now can type in your BASIC program. When you are ready to SAVE it, just SAVE it to tape as you would normally SAVE it. To VARIFY, rewind the tape to the start of the program, and type in, in direct commanded.

```
RAND USR 31744
```

And press play on your recorder and ENTER on your computer

If you saved the program with variables, then CLEARed them before verifying, or changed the variables in any way then you may get a verify error (R/0). Otherwise, if all is well, you'll get an 0/0 message to tell you that your program has been VERIFIED.

Reprinted from the Nov/Dec 87 issue of Ramtop of Greater Cleveland by way of Timelinez of the San Fran Bay Groups.



VSUG

The Vancouver Sinclair Users Group has been in existence since 1982. We are a support group for the owners and users of all SINCLAIR and TIMEX computers.

Pres:- Gerd Breunung PH#(604) 931-5509
V/Pres:- Glenn Read
Sec:- Harvey Taylor
Treas. & N/L Editor:- Rod Humphreys

Our membership dues are only $15.00/year and may be sent to the Treasurer:

Rod Humphreys
2006 Highview Place
Port Moody, B.C., V3H 1N5

Members of VSUG receive a monthly issue of ZXAppeal - our newsletter.

ZXAppeal accepts advertising. Our **PREPAID** rates are:

$10.00 – full page
$8.00 – 1/2 page
$5.00 – 1/4 page

ZXAppeal is distributed to approx 30 other SINCLAIR User Groups throughout North America as well as overseas via the NETWORK. NETWORK correspondence may be directed to the Editor at the above address.